Express Mail Label No. EL68025142US 09-18-00 A

UTILITY PATENT APPLICATION TRANSMITTAL
(Large Entity)

(Only for new nonprovisional applications under 37 CFR 1.53(b))

Docket No.
13807(YOR920000457US1)
Total Pages in this Submission
3

jc490 U.S. PTO 09/15/00

TO THE ASSISTANT COMMISSIONER FOR PATENTS

Box Patent Application
Washington, D.C. 20231

jc917 U.S. PTO 09/663664 09/15/00

Transmitted herewith for filing under 35 U.S.C. 111(a) and 37 C.F.R. 1.53(b) is a new utility patent application for an invention entitled:

USING TRUSTED CO-SERVERS TO ENHANCE SECURITY OF WEB INTERACTION

and invented by:

1. David M. Chess, 2. Joan Dyer, 3. Naomaru Itoi, 4. Jeff Kravitz, 5. Elaine Rivette Palmer, 6. Ronald Perez, 7. Sean William Smith

If a CONTINUATION APPLICATION, *check appropriate box and supply the requisite information:*

☐ **Continuation** ☐ **Divisional** ☐ **Continuation-in-part (CIP)** of prior application No.: ______

Which is a:

☐ **Continuation** ☐ **Divisional** ☐ **Continuation-in-part (CIP)** of prior application No.: ______

Which is a:

☐ **Continuation** ☐ **Divisional** ☐ **Continuation-in-part (CIP)** of prior application No.: ______

Enclosed are:

Application Elements

1. ☒ Filing fee as calculated and transmitted as described below

2. ☒ Specification having 38 pages and including the following:

a. ☒ Descriptive Title of the Invention

b. ☐ Cross References to Related Applications *(if applicable)*

c. ☐ Statement Regarding Federally-sponsored Research/Development *(if applicable)*

d. ☐ Reference to Microfiche Appendix *(if applicable)*

e. ☒ Background of the Invention

f. ☒ Brief Summary of the Invention

g. ☒ Brief Description of the Drawings *(if drawings filed)*

h. ☒ Detailed Description

i. ☒ Claim(s) as Classified Below

j. ☒ Abstract of the Disclosure

UTILITY PATENT APPLICATION TRANSMITTAL (Large Entity) *(Only for new nonprovisional applications under 37 CFR 1.53(b))*	Docket No. **13807(YOR920000457US1)**
	Total Pages in this Submission **3**

Application Elements (Continued)

3. ☒ Drawing(s) *(when necessary as prescribed by 35 USC 113)*
   - a. ☒ Formal   Number of Sheets **13**
   - b. ☐ Informal   Number of Sheets ______

4. ☒ Oath or Declaration
   - a. ☐ Newly executed *(original or copy)*   ☐ Unexecuted
   - b. ☐ Copy from a prior application (37 CFR 1.63(d)) *(for continuation/divisional application only)*
   - c. ☒ With Power of Attorney   ☐ Without Power of Attorney
   - d. ☐ *DELETION OF INVENTOR(S)*
     Signed statement attached deleting inventor(s) named in the prior application, see 37 C.F.R. 1.63(d)(2) and 1.33(b).

5. ☐ Incorporation By Reference *(usable if Box 4b is checked)*
   The entire disclosure of the prior application, from which a copy of the oath or declaration is supplied under Box 4b, is considered as being part of the disclosure of the accompanying application and is hereby incorporated by reference therein.

6. ☐ Computer Program in Microfiche *(Appendix)*

7. ☐ Nucleotide and/or Amino Acid Sequence Submission *(if applicable, all must be included)*
   - a. ☐ Paper Copy
   - b. ☐ Computer Readable Copy *(identical to computer copy)*
   - c. ☐ Statement Verifying Identical Paper and Computer Readable Copy

Accompanying Application Parts

8. ☐ Assignment Papers *(cover sheet & document(s))*

9. ☐ 37 CFR 3.73(B) Statement *(when there is an assignee)*

10. ☐ English Translation Document *(if applicable)*

11. ☒ Information Disclosure Statement/PTO-1449   ☒ Copies of IDS Citations

12. ☐ Preliminary Amendment

13. ☒ Acknowledgment postcard

14. ☒ Certificate of Mailing

    ☐ First Class   ☒ Express Mail *(Specify Label No.)*: **EL680251942US**

UTILITY PATENT APPLICATION TRANSMITTAL (Large Entity)

(Only for new nonprovisional applications under 37 CFR 1.53(b))

Docket No.
13807(YOR920000457US1)

Total Pages in this Submission
3

Accompanying Application Parts (Continued)

15. ☐ Certified Copy of Priority Document(s) *(if foreign priority is claimed)*

16. ☐ Additional Enclosures *(please identify below):*

Fee Calculation and Transmittal

CLAIMS AS FILED

For	#Filed	#Allowed	#Extra		Rate	Fee
Total Claims	42	- 20 =	22	x	$18.00	$396.00
Indep. Claims	6	- 3 =	3	x	$78.00	$234.00
Multiple Dependent Claims (check if applicable) ☐						$0.00
					BASIC FEE	$690.00
OTHER FEE *(specify purpose)*						$0.00
					TOTAL FILING FEE	$1,320.00

☐ A check in the amount of to cover the filing fee is enclosed.

☒ The Commissioner is hereby authorized to charge and credit Deposit Account No. **50-0510/IBM** as described below. A duplicate copy of this sheet is enclosed.

- ☒ Charge the amount of **$1,320.00** as filing fee.
- ☒ Credit any overpayment.
- ☒ Charge any additional filing fees required under 37 C.F.R. 1.16 and 1.17.
- ☐ Charge the issue fee set in 37 C.F.R. 1.18 at the mailing of the Notice of Allowance, pursuant to 37 C.F.R. 1.311(b).

Signature

Dated: **September 15, 2000**

Richard L. Catania
Registration No. 32,608

SCULLY, SCOTT, MURPHY & PRESSER
400 Garden City Plaza
Garden City, New York 11530
(516)742-4343

cc:

USING TRUSTED CO-SERVERS TO ENHANCE SECURITY OF WEB INTERACTION

Background Of The Invention

This invention generally relates to transactions using the World Wide Web; and more specifically, the invention relates to improving the security of such transactions.

The World Wide Web is the grounds where, on a broad scale, our society's business, government, and personal services are migrating and growing. As a basic model, a large population of clients with browsers obtain services from a smaller population of service providers operating Web servers. However, for each critical service that takes root in the Web (and arguably for many purely recreational services as well), the financial and personal interests of the clients force them to trust the integrity and privacy of the computation and data storage at the service providers.

Distributed computation (and even centralized computation, with multiple parties) introduces a fundamental problem: distribution dissociates dependency from control. Consider a basic scenario outlined in Figure 1. In this example, Alice and Bob participate in some computational activity. Alice's interests I depends on some correctness and/or privacy properties P of some computation X at a computer that Bob controls. Consequently, Alice must depend on Bob to preserve and protect her interests. However, Bob may have no motivation to do this; and, in fact, Bob's interests may

conflict with Alice's, and motivate him to actively subvert Alice's computation.

In the above example, dependency on remote computation went one way. However, the scenario can be more complex, as Figure 2 shows. In this example, suppose Alice and Bob are users in a decentralized e-cash system, where cash is a value in a register in a wallet, and is exchanged by a protocol between the wallets. The computations X_A, X_B are the storage and appropriate alteration of the amount of money in Alice's wallet and Bob's wallet, respectively. The important security properties P_A, P_B of these computations are that the values in these wallets only increase under appropriate circumstances. Alice's interests I_A include maximizing the amount of money she has, and preserving its value; Bob's interests I_B are symmetric.

If Alice can break into her wallet, she can break P_A; similarly, Bob can break P_B. Alice's interests I_A depend on P_B holding; but Bob's interests I_B motivate him to break P_B. Symmetrically, Bob's interests depend on P_A, which Alice is motivated to break.

All parties in this distributed e-cash system must trust all other parties; in a sense, the least-trusted user has the ability and the motivation to subvert the entire system.

Previous research had long speculated that programmable, trusted secure coprocessors could enable systematic solutions to problems such as Figure 1. Figure 3

illustrates a revised scenario. If X occurred in a secure coprocessor at Bob's machine, and Alice was able to authenticate that X was occurring there, beyond Bob's control, and Bob's ability to manipulate his host and its network connections could not subvert P, then Alice can trust that the important properties P_B still hold of X, despite Bob's potential attacks.

As the popularity of the Web---and the recognition of its potential for applications with real security issues---spread, many proposals and ideas surfaced to add security to the basic http protocol. At one point, three primary contenders emerged:

i) Shen from CERN,

ii) Secure HTTP from a consortium including NCSA, and

iii) Secure Socket Layer (SSL), from Netscape.

Primarily because Netscape's SSL protocol was the first to be widely deployed, SSL became the de facto standard for securing Web transactions.

As practiced, SSL permits the client to establish a shared symmetric key with a specific authenticated server. The server has a private-public keypair, and a certificate from some CA attesting to certain properties about the entity owning this public key. The client browser has some notion of which CA root keys it recognizes as valid. When a client opens an SSL connection, it verifies that the certificate from the server is correctly signed by a CA root that the client's browser currently recognizes as legitimate. The client

and server then carry out a key generation/exchange protocol that ensures that the client, and a party which knows the private key matching the server's public key, share a symmetric key---that is (theoretically) shared by no one else, not even an adversary observing the messages between the client and server.

The remainder of the SSL session is then encrypted with this session key. Encryption with a key obtained this way provides several properties. Both parties can trust the privacy of data from the client to the server. Both parties can trust the privacy of data from the server back to the client. Both parties can trust that an adversary cannot alter or manipulate data in either direction without detection (since SSL provides integrity checking and sequence numbering). The client can trust the authenticity of the server (since the server entity must know the private key matching the public key in the certificate). The server can trust that, throughout the session, the entity claiming to be the client is the same entity that started the session. Figure 4 shows a more detailed ladder diagram.

Even with the current state of deployed technology (i.e., SSL), however, all the client can be sure of is the identity of the entity who originally possessed the public key in that server's certificate.

At best, this identity establishes good intentions---if the alleged service provider has a pre-existing reputation that makes this hypothesis plausible. On the other hand, a service provider with an unknown reputation

might be downright malicious. Also, any service provider may have good intentions, but may be careless with general site security. Moreover, the entity with which the client is currently interacting may not even be this original service provider, but rather an imposter who has learned the private key.

The threat that arises from this uncertainty is amplified by the Web's distribution of computation from server to client: via Java and Javascript, and also via more subtly executable content, such as Word documents infected with Macro viruses. Furthermore, many interactions involve more parties than just the client and server, but these additional parties are also forced to trust the server integrity.

This situation---that participants are forced to trust server integrity, but have no basis for this trust---is a fundamental problem threatening a wide variety of Web applications. Several of these applications are discussed below. These applications are shown herein to represent examples having missing security and/or privacy properties.

AUTHENTICATION OF CLIENTS

The current Web infrastructure prevents a server from being able to prove anything to a third party about the identity of an alleged client. Without a public-key infrastructure for citizens, clients are forced to use human-usable authenticators, such as user ids and passwords. However, in the current infrastructure, these

are exposed to the server of unknown integrity. As a consequence of this exposure, an adversary who compromises the server (or a malicious server operator) can impersonate this user at that site and any others where the client has used that password. This exposure also prevents legitimate server operators from being able to argue it really was a particular client who opened a particular a session. In this application, "user" and "client" are used interchangeably.

NONREPUDIATION OF CLIENT ACTIVITY

The current Web infrastructure prevents a server from being able to prove anything to a third party about the activity of an alleged client. For example, how can an insurance company taking an application over the Web turn around and prove that a particular individual really answered that question that way?

NONREPUDIATION OF SERVER ACTIVITY

The current Web infrastructure prevents a server from being able to prove anything to a third party about the activity of the server---including the questions that generated the answers a client provided.

CREDIT CARD TRANSACTION SECURITY

The current Web infrastructure provides secure transmission of a client's credit-card information and transaction amount to a server, where they are then exposed. An adversary who compromises this server (or a

malicious server operator) can change the amount of the transaction, retain the amount but repeat the transaction many times, or use the credit card information to forge additional transactions. This situation may significantly reduce the potential market for new e-merchants without a pre-established reputation.

TAXES ON E-COMMERCE ACTIVITY.

The current Web infrastructure provides no acceptable means for a third party with legitimate interests (such as a government's tax collection service) to accurately learn certain information about individual or collective Web interactions (such as how much sales tax an e-merchant owes them for last month). Reporting all transactions to the government would be unacceptable to the merchant and customer for privacy concerns; while reporting only a total amount owed would be unacceptable to the government, since the figure would be unverifiable, and the merchant reporting this unverifiable figure would be motivated to understate it.

RE-SELLING OF INTELLECTUAL PROPERTY

The current Web infrastructure provides no acceptable means for a third party who participates in an interaction indirectly, by licensing proprietary information to the server, to protect their legitimate interests. For example, a publisher who owns a large copyrighted image database might wish to make this available to a university library---but might worry that

compromise of the university server will compromise the database.

PRIVACY OF SENSITIVE OR PROPRIETARY WEB ACTIVITY

The current Web infrastructure provides no means for a server operator to plausibly deny that they (or an adversary who has compromised their machine) is not monitoring all client interactions. How can companies that are accessing a competitor's server, know for sure that said competitor is not data-mining their queries? What about people who wish to purchase sensitive literature (about health topics, or currently unfashionable politics)? If an auction server provides a bulletin board service where customers can post "anonymous, confidential" comments, how do the customers know their identity is being kept secret? What about a server who is participating in an anonymous re-rerouting service?

CORRECTNESS OF WEB ACTIVITY

The current Web infrastructure provides no means for a server operator to establish that they (or an adversary who has compromised their machine) has not otherwise altered or corrupted important correctness properties of the service. In the auction bulletin board service described above, how can customers know that the anonymous posts came from bona fide customers, and that the timestamps are correct?

ENFORCEMENT OF LOGO/"SEAL OF APPROVAL" LICENSES

The current Web infrastructure provides no effective means for a party to ensure that logos or endorsements appear only on the appropriate server pages. For example, a company could establish an "inspected" logo to endorse servers who have withstood inspection by the ethical hackers of IBM Global Services. However, any client who visits these pages can capture the logo, and put it on any page.

SAFETY OF DOWNLOADABLE CONTENT

The current Web infrastructure provides no means for the client to ensure that executable content downloaded from a server is indeed safe, short of the client themselves actually running the latest anti-virus software. Since most consumers do not do this, this leaves them at risk. Moving this computation (and the anti-virus update problem) to the server is more efficient---but how can clients know the server really carried this out?

AUTHENTICITY OF DOWNLOADABLE CONTENT

The current web infrastructure provides no easy means for the client to authenticate the origin of downloadable content. Posters of content can provide digital signatures, but then the client needs to explicitly obtain and verify the trust chain on each item. Moving this computation (and the latest certificate revocation lists) to the server is more efficient---but how can clients know the server really carried this out?

INTEGRITY OF SERVER MACHINE

The current Web infrastructure provides no means for the client to recognize those servers whose hosts do run more secure operating systems or have more secure administrative practices. How can a consumer know for sure that a site really ran a particular network security analyzer or used a particular new secure boot system?

Summary Of The Invention

An object of this invention is to provide a way for parties in a Web interaction to have confidence in the server integrity.

Another object of the present invention is to add a secure coprocessor to an existing service provider infrastructure.

A further object of this invention is to provide a set of programs for a coprocessor for an existing service provider, that address the fundamental web security problem by raising the trust level of the computation and data storage at the server.

Another object is to provide these properties without substantial changes to the client infrastructure.

Another one: that a server operator can enhance his service to have these properties, by adding hardware and software to his own site (instead of, for example, moving

computation to a literal third party somewhere else in the net).

These and other objects are attained with, and with a method of using, a trusted co-server for a service provider. The co-server executes a program such that:

for multiple parties P_0-P_n (where P_o is said co-server),

each party P_i may (optionally) provide input I_i,

and then said co-server carries out N + 1 functions: F_i $(I_o...I_n)$ describes what the co-server returns to party P_i.

The preferred embodiment of the invention, as described below in detail, raises the trust level of the computation and data storage at the sever. For instance, this invention may be witness to authenticity of certain data coming back to the client. This data can include assertions from the trusted guardian about the server content and configuration. We use the term "guardian" to refer to the trusted co-server. The invention, also, can provide privacy of data going back to the server, by keeping it encrypted between the client and the guardian, and then re-encrypting it before inserting it into the server.

With this invention, the user can trust the integrity of the computation occurring at the guardian---even if the server operator might be motivated to subvert it. The guardian also provides a trusted haven for computation

relevant to third parties who may also have an interest in the client-server interaction.

As used herein a co-server is another computer participating in the service. A co-server is trusted (referred to as a trusted co-server) when the client and/or server operator can trust that this co-server operates securely. A secure coprocessor is a computer with sufficient physical and logical security protections so that it can be trusted to carry out its computation despite attack by an adversary with direct physical access. The IBM 4758 (further discussed in "building a high-performance, programmable secure coprocessor," by Smith and Weingart, Computer Networks 31 (1999) 831-860) is an exemplary secure coprocessor; withstanding Level 4 validation against the FIPS 140-1 standard is an exemplary way of establishing that a coprocessor has sufficient physical and logical security protections.

Other methods may discuss using secure coprocessors as accelerators of SSL connections in the Web sites, but not using these as a trusted third party participating in the interaction in accordance with the present invention. For example, in the other methods the symmetric key guarding the client session is known by the server. Thus any communication sent back and forth is known by the server, thus forfeiting the security and privacy advantages provided by the present invention.

An aspect of the present invention is to provide an advantageous, (and most often relatively painless) way for clients to establish an authenticated and private

channel to a trusted co-server. This is advantageously performed with minimal change to the current client infrastructure.

As example of a useful embodiment is when the service is a Web service, and a relatively "painless way" is SSL. Those familiar with the art will realize the many varying ways to use this trusted third party for various types of sessions and applications.

As used herein the term operator includes any of the many different types of operators. For example an "operator of service" may rent space on someone else's server. In this case, the "operator" may refer to said service operator, or said server operator.

The present invention is adaptable to a service and more particularly to a computational service. As used herein, a computational service is a service whose provision involves a computer. Examples include any information --and/or data-- provider such as received and/or exchanged with a Web site, and especially an information/data-only Web site, and also a Web or other site through which a user purchases a physical object, etc.

One embodiment of the invention is a method for enhancing a service to provide security and privacy to each client of a plurality of clients. Said service includes computation. An exemplary service might be a Web site, with the clients being the remote users of this site accessing it via browsers. The invention moves a

selected portion of the computation from a server into a trusted co-server executing to interact with the server through the co-server. In some embodiments the portion is the entire computation.

In another embodiment of this invention, the step of moving and enabling include providing a trusted third party at the server. That is, the client and/or server can trust the co-server to operate securely despite potential efforts by the client and/or server to compromise this security.

In another embodiment of this invention, the step of allowing includes enabling the client to have an authenticated, private channel to the co-server.

In another embodiment of this invention, the service is a Web service and the clients are remote users operating browsers.

In another embodiment of this invention, the step of enabling includes the client using the co-server's certified keypair to establish a shared symmetric key.

In another embodiment, the step of enabling includes using the Secure Sockets Layer (SSL) protocol.

Further benefits and advantages of the invention will become apparent from a consideration of the following detailed description, given with reference to the accompanying drawings, which specify and show preferred embodiments of the invention.

Brief Description Of The Drawings

Figure 1 illustrates a basic scenario in which a Web user depends on a server to protect the user's interests.

Figure 2 depicts a mutual trust scenario in which people depend on each other to protect their respective interests.

Figure 3 shows a revision of the scenario of Figure 1, in which a secure coprocessor is used to protect the interests of the user.

Figure 4 is a ladder diagram illustrating a Web security protocol referred to as Secure Socket Layer (SSL).

Figure 5 shows the hardware of an exemplary secure coprocessor platform embodying this invention.

Figure 6 shows the software configuration architecture for an exemplary coprocessor platform.

Figure 7 shows an exemplary process for a server operator to use a secure coprocessor platform to install and certify a trusted co-server.

Figure 8 shows an exemplary establishment of secure SSL session between client and trusted co-server.

Figures 9-13 show some various ways in which a Web application can use a trusted co-server to enhance security and privacy.

Detailed Description Of The Preferred Embodiments

Figure 5 shows the hardware of an exemplary secure coprocessor platform, based on the commercially available IBM 4758 Model 2. The device provides general-purpose computing environment for applications (502, 503, 504, 505), with hardware support for cryptographic applications (507, 508, 509, 510). However, the device also provides crucial security features, including

i) Continuously active tamper-detection circuitry (501) monitors tamper detectors (513) and, in case of physical attack, destroys sensitive secrets in secure memory (503, 504) before an adversary can access them; and

ii) Hardware locks (506) protect crucial code and secrets from possibly malicious or faulty application code, preserving the ability of each device to properly authenticate its configuration, and preventing a device with a rogue application from impersonating other devices and applications.

Figure 6 shows the software configuration architecture for an exemplary secure coprocessor platform, based on the commercially available IBM 4758 Model 2. The coprocessor vendor (601) gives an application developer (602) a unique identifier, a signed command telling coprocessors with no application to recognize that

developer as their application owner, and a signed command telling coprocessors that that application developer has a specified public key.

The application developer (602) then signs his application code with his private key, and gives this signed code, along with the vendor-provided commands, to the user (603). The user (603) provides these items to the security configuration software (605) within the secure coprocessor (604). This software validates the commands against the vendor's public key and other parameters in the parameters store (606). If things validate, the security configuration software takes these steps:

i) it updates the parameter store (606) to record that the application developer (602) now owns the application space within this device, and records the developer's idea and public key,

ii) it installs the application as the device's application software (607),

iii) it generates a keypair (609) for this application installation on this device; uses the devices's own keypair (608) to certify that this new keypair belongs to that application, for that owner, in that device; and leaves this application keypair (609) in a place where the application software (607) can access it at run-time.

We note that Figure 6 shows an exemplary architecture only. Coprocessors with architectures that provide for a

layer of system software below the application software (such as the current IBM 4758) can be configured to provide the important properties of Figure 6.

Figure 7 shows an exemplary process for a server operator to use a secure coprocessor platform to install and certify a trusted co-server. The server operator (701) obtains a secure coprocessor platform (703), and uses the mechanisms of Figure 6 (e.g., 704, 705, 707) to install co-server application software (706) from a co-server software vendor (702) into this device. The co-server application then generates another keypair (709). The server operator uses the co-server application's ability to authenticate itself with the co-server keypair (708), to prove to the satisfaction of a recognized SSL certificate authority (712) that said new keypair (709) belongs to an installation of said co-server application (706) securely running on an untampered secure coprocessor platform (703) at the site of said server operator (701).

The SSL Certificate Authority then issues an SSL-compatible certificate attesting to the public key of this keypair (709) and the entity (co-server application inside secure coprocessor at server operator) to which it belongs. The co-server application stores this certificate, and is then ready to participate as a trusted co-server to server operator's web application (711) on his web server (710).

Figure 8 shows an exemplary establishment of secure SSL session between client and trusted co-server. A remote

client (807) using a Web browser (808) initiates an SSL session with the co-server application (803) within the secure coprocessor (802) at the web site maintained by this server operator (801). Because client's web browser (808) indicates that the co-server application (803) suitably demonstrates knowledge of the private key matching the public key in this application's SSL-certified keypair (804), the client (807) can reasonably conclude that server-client communications within this SSL session originated within the trusted co-server (802, 803) and that client-server communications terminate in the trusted co-server (902,803)---even if the server operator (801) may be motivated to maliciously alter or spy on these communications.

Figures 9 through 13 show some various ways in which a web application can use a trusted co-server to enhance security and privacy.

Figure 9 shows how a client can engage in a session with an insecure server (901), agree on a price for a product (902), then open an SSL session (903) to a trusted co-server, configured with a payment application. The server forwards the price to the co-server (904), which displays this and accepts the client's private credit card information (905) and signs and encrypts the pair (with a serial number, to prevent replay) (906). The server operator can then inject this signed encrypted packet into the payment system (907.)

This application ensures:

i) that client's private information remains private even from the server operator, and

ii) the client's credit card is only charged once, and for the agreed-on amount, even if the server operator (or a hacker who has compromised the server) attempts to cheat.

Figure 10 shows how a client can open his interaction by establishing an SSL session (1001) with a trusted co-server configured with a server status application. The co-server displays some authenticated information to the client (1002) (such as: the security status and appropriate logos or seals of approval) about the server, and provides a link by which the user can click to proceed to the server (1003). (Following this link terminates the SSL session.) This ensures that the client gets accurate information about the server---even if the server operator might be motivated to falsify this information. For additional security, the co-server could assist in establishing a new SSL session for the client when interacting directly with the server.

Figure 11 shows how a client can open his interaction by establishing an SSL session with a trusted co-server (1101) configured with an authentication application. The co-server prompts the client (1102) for client authentication information, such as a user id and password. The client responds (1103), and the co-server verifies this information (1104), and then directs the client to the main web server (1105) but also provides

this server with an authentication token indicating that the client has properly authenticated (1106). The SSL session then ends, and the client then interacts with the main server (which requires such a token to function) (1107). This protects the security of a restricted application, while also protecting the privacy of the client's authenticators from a malicious server operator or a compromised web server.

Figure 12 shows how a client can open an SSL session (1201) with a trusted co-server configured with a private information retrieval application. The client requests (1202) a particular page of private data; the co-server then uses private information retrieval techniques (1203) to obtain this page from the set of pages stored on the main server, in such a way that the server operator learns no information about which page was requested.

The co-server then decrypts this page (1204), and returns the plaintext to the user through the encrypted SSL channel (1205). This ensures that the client can obtain information such as potentially embarrassing medical data without revealing the data to a malicious server operator or compromised server. This would also ensure that a server operator could not be forced to reveal which data which clients are examining.

Figure 13 shows how a client can open an SSL session (1301) with a trusted co-server configured with a filter application. (Such a filter might be a virus-scanner, for example.) The co-server (at the request of the client) then forwards client queries to the web server

(1302). The co-server intercepts the server response (1303), runs the filter on them (1304), and packages the two into a page such that the co-server's response is in one well-defined portion of the page, and the server's response is confined to the other (1305). The co-server then sends this package back to the client via the SSL channel (1306). This ensures that the client can interact with the server---but gets authenticated filter output on each page from the co-server, even if the server operator might be motivated to falsify this output.

This invention can address each of the example problems discussed above: by enhancing a service to provide one or more desirable levels of security and/or privacy properties. This includes properties described herein and other properties known to those skilled in the art. A desirable level includes a level desired by a client, a service, a third party (e.g. a bank, a library, a data provider, a web site, a seller, a trusted authority, an operator, a manager, etc.) and any combination of these. Thus in an example embodiment the present invention provides methods and apparatus for a service to provide a client with a desired security level. This is advantageous particularly in applications missing security and/or privacy properties. As used herein the term security uses a broad definition to include, but not be limited to, correctness, non-disruption, overcoming and/or reacting to real and/or potential adversarial actions, maintaining all facets of privacy, etc.

AUTHENTICATION OF CLIENTS

The program at the co-server can trap the password, authenticate the client, then issue a signed receipt that that client properly authenticated for that session.

NONREPUDIATION OF CLIENT ACTIVITY

Besides issuing a receipt for client identity, the program at the co-server can issue a signed receipt for the entire transaction.

NONREPUDIATION OF SERVER ACTIVITY

The co-server can issue a receipt for the entire transaction, including the prompts the server provided, which generated the answers the client provided.

CREDIT CARD TRANSACTION SECURITY

The program at the co-server can trap the credit card and transaction information, and inject it directly into the acquirer's system. The credit card number data never appears in plaintext at the server site; the server operator or a penetrator has no opportunity to inflate the transaction amount; and (unlike SET) the client need not change the way they operate. (This co-server application could be included as part of an entry-level e-merchant start-up package.)

TAXES ON E-COMMERCE ACTIVITY

The program at the co-server can monitor the total tax owed by that merchant for the transactions that went

through that co-server (e.g., because of some other co-server application there), and report that authenticated total back to the governmental entity.

RE-SELLING OF INTELLECTUAL PROPERTY

The owner of the property could provide it in ciphertext to the server; the co-server would decrypt the particular items being used, and ensure that appropriate licensing/royalty/watermarking requirements were being enforced.

PRIVACY OF SENSITIVE OR PROPRIETARY WEB ACTIVITY

With the proper private information retrieval scheme on the back-end, the co-server can fetch and provide the content to the client, without the server operator being aware of the content being fetched.

CORRECTNESS OF WEB ACTIVITY

The computation critical to the appropriate correctness properties can be moved into the co-server---whose application program would need to advertise that it was performing these computations.

ENFORCEMENT OF LOGO/"SEAL OF APPROVAL" LICENSES

The logo information could be provided, when appropriate, by the trusted co-server; logos that did not appear on an authenticated co-server-to-client channel are not legitimate.

SAFETY OF DOWNLOADABLE CONTENT

An application at the co-server could run the latest anti-virus software either dynamically, as the data was being downloaded, or off-line (but then cryptographically verifying that the data being downloaded had indeed been scanned earlier). Clients can trust that content downloaded via this SSL-authenticated channel from the trusted co-server has been scanned.

AUTHENTICITY OF DOWNLOADABLE CONTENT

An application at the co-server can verify the signatures of the posted content. Clients can trust that content downloaded via this SSL-authenticated channel from the trusted co-server has been authenticated. (Indeed, the client only need download the identity of the poster, not his public key, signature, and appropriate certificates.)

INTEGRITY OF SERVER MACHINE

If the trusted co-server can witness that the appropriate computational security tool (such as a network security analyzer or secure booting technique) was applied to the host---perhaps because this tool was applied from the co-server itself, or from a companion trusted machine---then it can include this data in the SSL-authenticated communication channel from the co-server to the client.

It should be noted that the above-discussed examples are exemplary embodiments, built around technology currently available commercially. Those skilled in the art would be able to develop alternate embodiments---particularly as new secure co-processing technology becomes available, and as continued experimentation and prototyping suggests modifications and improvements.

While it is apparent that the invention herein disclosed is well calculated to fulfill the objects stated above, it will be appreciated that numerous modifications and embodiments may be devised by those skilled in the art, and it is intended that the appended claims cover all such modifications and embodiments as fall within the true spirit and scope of the present invention.

CLAIMS

1. A method, comprised of enhancing a computational service to each client of a plurality of clients, by:
moving a selected portion of a computation from a server into a trusted co-server executing inside a secure coprocessor; and
allowing each client to interact with the server and the co-server.

2. A method as in Claim 1, wherein the step of allowing includes providing a trusted third party at said server.

3. A method as recited in Claim 1, wherein said step of allowing includes enabling said client an authenticated, private channel to said co-server.

4. A method as in Claim 1, wherein said service is a Web service and said clients are remote users operating browsers.

5. A method as in Claim 3, wherein said step of enabling includes the client using the co-server's certified keypair to establish a shared symmetric key.

6. A method as in Claim 5, wherein said step of enabling includes employing the Secure Sockets Layer (SSL) protocol.

7. A method as in Claim 1, wherein said step of moving includes integrating functions of said co-server in a same machine as said server.

8. A method as in Claim 1, wherein said step of enhancing includes providing a desired security and/or privacy property.

9. A method as in Claim 1, wherein said step of enhancing includes providing at least one security and/or privacy property to an application selected from the group including: authentication of clients, nonrepudiation of client activity, nonrepudiation of server activity, credit card transaction security, taxes on e-commerce activity, re-selling of intellectual property, privacy of sensitive or proprietary web activity, correctness of web activity, enforcement of logo and/or "seal of approval" licenses, safety of downloadable content, authenticity of downloadable content, integrity of server machine, and any combination of these.

10. A method as in Claim 1, wherein:
input from said client is prompt from server for the user's private authenticator data, such as a password, input from said server is this authentication data, co-server algorithm that generates output to said client based on said current co-server state and said inputs indicates whether or not the authenticator data is correct for this user.

11. A method as in Claim 1, where co-server algorithm that generates output to said server based on said current co-server state and said inputs includes a signed statement, using a private key known to the co-server,

attesting, for the server, that the client engaged in an interaction satisfying certain properties.

12. A method as in Claim 1, where co-server algorithm that generates output to said client based on said current co-server state and said inputs includes a signed statement, using a privacy key known to the co-server, attesting, for the client, that the server engaged in an interaction satisfying certain properties.

13. A method as in Claim 1, wherein:
the client's input includes a credit card number (CCN), the output co-server algorithm that generates output to said client based on said current co-server state and said inputs includes the CCN, encrypted so that the server cannot read it but an acquirer can.

14. A method as in Claim 13, wherein:
the server's input includes a transaction amount, the output co-server algorithm that generates output to said client based on said current co-server state and said inputs includes the transaction amount, cryptographically bound to the encrypted CCN so that the server cannot alter it.

15. A method as in Claim 1, where:
the client's input includes a credit card number,
the server's input includes a transaction amount,
the co-server encrypts this CCN so that the server cannot read it but an acquirer can, and cryptographically binds the transaction amount to the this encrypted CCN, then, at some point during or after the interaction, transmits

this data to the acquirer in such a manner so that the acquirer can receive this transaction exactly once.

16. A method as in Claim 1, wherein: the interaction via the server input and/or the client input, includes a transaction amount A, the co-server input may include an accumulated total, the function co-server algorithm that generates new co-server state based on said current co-server state and said inputs updates the accumulated amount by adding T(A), where T is a predefined function, such as: a map from an amount to the taxes owed on that amount, and at some point during or after this interaction, the co-server produces an authenticated statement of the current value of the accumulated amount.

17. A method as in Claim 1, where: a remote party is an owner of intellectual property, the server input includes part of this property, encrypted so that only the co-server can decrypt it, the output function co-server algorithm that generates output to said client based on said current co-server state and said inputs to the client includes a portion of the decryption of input from said client.

18. A method as in Claim 17, except the output function co-server algorithm that generates output to said client based on said current co-server state and said inputs now includes a transformation of a portion of the decryption of input from said server, where said transformation may include adding a watermark.

19. A method as in Claim 17, except the output function now includes a transformation of a portion of the decryption of input from said server, where said transformation may include reducing the quality of the plaintext.

20. A method as in Claim 17, except the output function now includes a portion of the decryption of input from said server, re-encrypted, possibly with rights management rules, in a manner that a secure coprocessor at the client site can decrypt it.

21. A method as in Claim 1, wherein:
the client input includes a choice of which record R in a set of records the client would like to receive, the co-server includes this record R in its response to the client, however, the co-server obtains R in such a way as the server does not know which record was the one selected.

22. A method as in Claim 1, wherein:
a remote party establishes a content evaluation scheme, consisting of an evaluation function mapping content to some set of indicators, and as part of computing the client output function co-server algorithm that generates output to said client based on said current co-server state and said inputs, the co-server calculates, or verifies an external calculation, of the evaluation function and includes the result in the client output.

23. A method as in Claim 22, where the evaluation function consists of determining whether specified server

input from specified server merits a logo or seal of approval, in accordance with a business arrangement between the server and the remote party.

24. A method as in Claim 22, where the evaluation function consists of determining whether server input which has potentially executable content is free of viruses.

25. A method as in Claim 24, where the evaluation function is parameterized by a "signature file" and where the client output includes an identification of which signature file was used in this interaction.

26. A method as in Claim 22, where party the remote party has injected evaluation function and/or some of its parameters into the co-server through a private channel, so that the server cannot know the details of the evaluation function execution occurring on the co-server.

27. A method as in Claim 22, where the server input includes both content and a signature on the content, from one of possibly many content providers, and the evaluation function includes testing whether the signature is valid.

28. A method as in Claim 1, where:
a remote party establishes a content evaluation scheme, consisting of an evaluation function mapping content to some set of indicators, and as part of computing the server output function co-server algorithm that generates output to said client based on said current co-server

state and said inputs or internal function co-server algorithm that generates new co-server state based on said current co-server state and said inputs the co-server calculates, or verifies an external calculation, of the evaluation function input from said client and includes the result in the output.

29. A method as in Claim 1, where:
the co-server has the ability to carry out security-enhancing actions against the server, such as booting the server and securely or carrying out a security scan of the server, the output returned to client indicates which of these actions have been carried out, and how recently.

30. A method as in Claim 1, where:
the client can specify whether the interaction is a read interaction or a write interaction;
for a write interaction:
the client input includes a message M and a specification S of the appropriate entities who can read this message;
the co-server retains M and S by storing them in some combination across the co-server and server via an algorithm that generates new co-server state based on said current co-server state and said inputs, the internal state in the co-server and co-server algorithm that generates output to said server based on said current co-server state and said inputs;
however in said write interaction:
any portion of M sent via co-server algorithm that generates output to said server based on said current co-

server state and said inputs is encrypted, so that the server cannot access the plaintext;
and mechanisms are used to ensure that, when the co-server later retrieves any of this data from the server, that the data has not been changed;
for a read interaction:
the client input specifies which message M the client would like to read, the co-server retrieves S; if the client satisfies S, then the co-server sends M back to the client, after first retrieving and decrypting it, if necessary.

31. A method for enhancing a service to provide security and/or privacy to each client from a plurality of clients, said service including computation on a server controlled by an operator, the method comprising:
moving a selected portion of said computation from a server controlled by said operator into a trusted co-server executing inside a secure coprocessor; and
allowing clients to interact with the server through the co-server.

32. A method as recited in Claim 31 wherein the secure coprocessor is installed at the server.

33. A method for enhancing a service including computation on a server controlled by an operator, the method comprising:
providing at least one security and privacy property to at least one client from a plurality of clients by:

moving a selected portion of said computation from a server controlled by said operator into a trusted co-server executing inside a secure coprocessor; and
enabling clients to interact with the server and the co-server.

34. A trusted co-server, executing a program such that:
for multiple parties, including a Web server a remote client and said co-server, each party may, optionally, provide input, and then the co-server carries out for each party, a function on all these inputs, and optionally returns output to said each party; and
wherein the co-server executes so that parties such as the remote client can authenticate and trust the correct execution of the co-server despite attempts by the Web server to subvert this.

35. A trusted co-server according to Claim 34, wherein the co-server executes inside a tamper respondent secure coprocessor.

36. A trusted co-server according to Claim 34, wherein the secure coprocessor is co-located at said server.

37. A method of enhancing the security of a Web based transaction utilizing a server, the method comprising the steps:
providing the server with a trusted co-server; and
using the trusted co-server to execute a program such that:
for multiple parties,

each party may, optionally, provide input and then said co-server carries out for each party, a function on all these inputs.

38. A method according to claim 37, where: one party is a Web server and another party is a remote client.

39. A method according to Claim 37, where: the client authenticates the co-server, the client sends its input to the co-server over a private channel, such as one established by encryption with a shared secret key, the co-server sends its output to said another party over a private channel, such as one established by encryption with a shared secret key.

40. A program storage device readable by machine, tangibly embodying a program of instructions executable by the machine to perform method steps for enhancing a computational service to at least one client of a plurality of clients, said method steps comprising: moving a selected portion of a computation from a server into a trusted co-server executing inside a secure coprocessor; and allowing each client to interact with the server and the co-server.

41. A program storage device according to Claim 40, wherein the step of allowing includes providing a trusted third party at said server.

42. A program storage device according to Claim 41, wherein the step of allowing includes enabling said client an authenticated, private channel to said co-server.

USING TRUSTED CO-SERVERS TO ENHANCE SECURITY OF WEB INTERACTION

ABSTRACT

A trusted co-server, and a method of using a trusted co-server, for a service provider. The co-server executes a program such that: for multiple parties P_0-P_n (where P_o is said co-server), each party P_i may (optionally) provide input I_i, and then said co-server carries out N functions: F_1 $(i_o \ldots I_n)$ describes what the co-server returns to party P_i. The preferred embodiment of the invention raises the trust level of the computation and data storage at the sever. For instance, this invention may be witness to authenticity of certain data coming back to the client. This data can include assertions from the trusted co-server about the server content and configuration. The invention, also, can provide privacy of data going back to the server, by keeping it encrypted between the client and the co-server, and then re-encrypting it before inserting it into the server. With this invention, the user can trust the integrity of the computation occurring at the co-server---even if the server operator might be motivated to subvert it. The co-server also provides a trusted haven for computation relevant to third parties who may also have an interest in the client-server interaction.

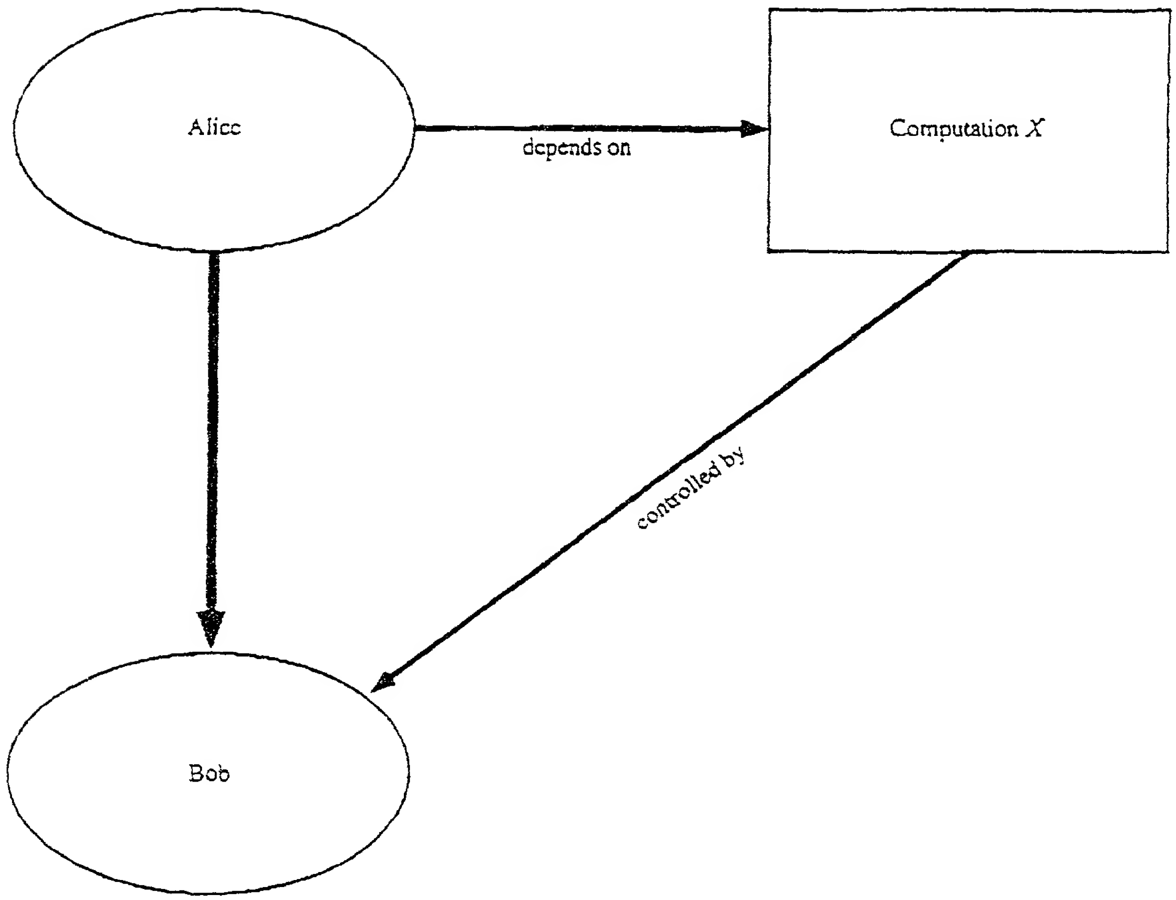

Figure 1 The basic motivating scenario: in distributed computation, one party (here, Alice) can incur a dependency on a party who may have different interests (here, Bob), because computation critical to the first party may reside on a machine the second party can control.

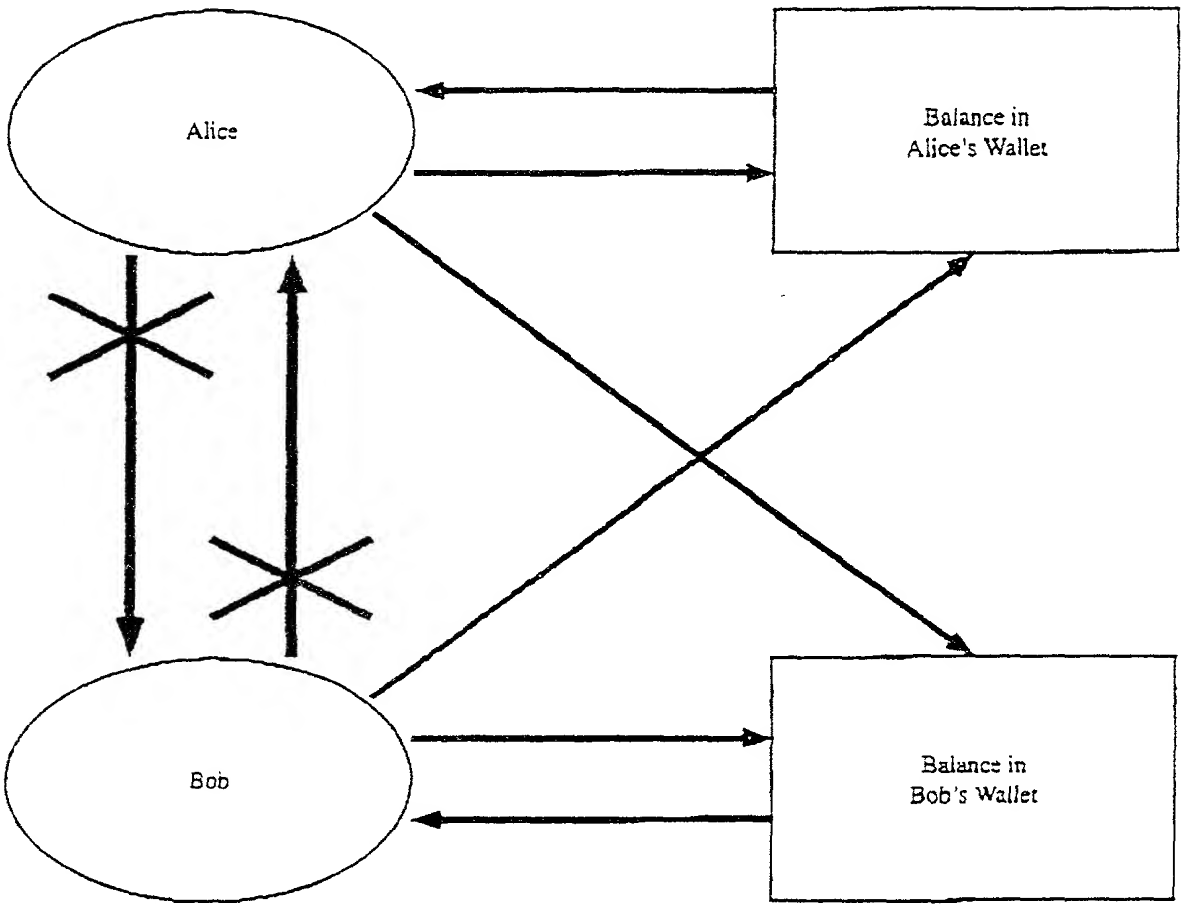

Figure 2 Applications like fully decentralized e-cash create scenarios where parties incur mutual dependency: preserving the interests of each one depends on the good graces of the other.

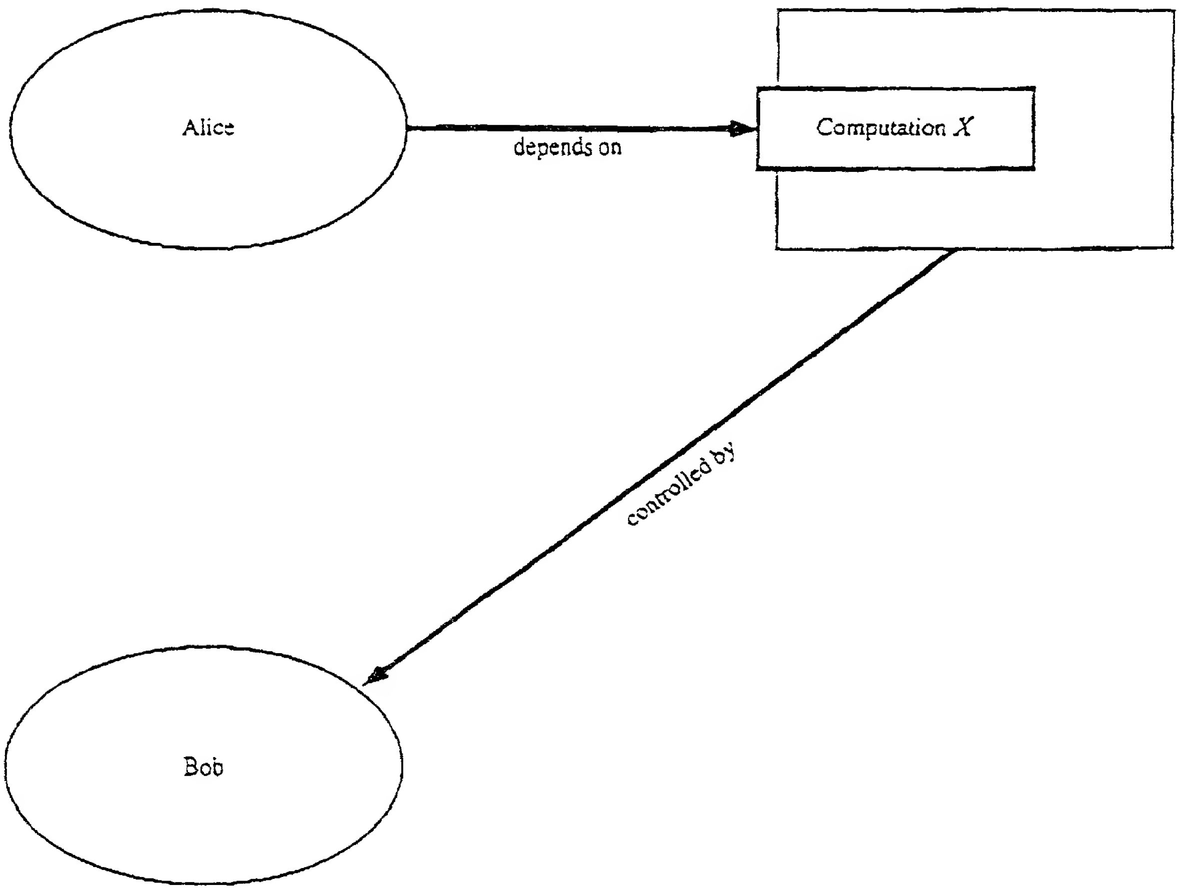

Figure 3 By moving the computation on which Alice depends from Bob's machine to a secure coprocessor added to Bob's machine, Alice no longer incurs a dependency on Bob.

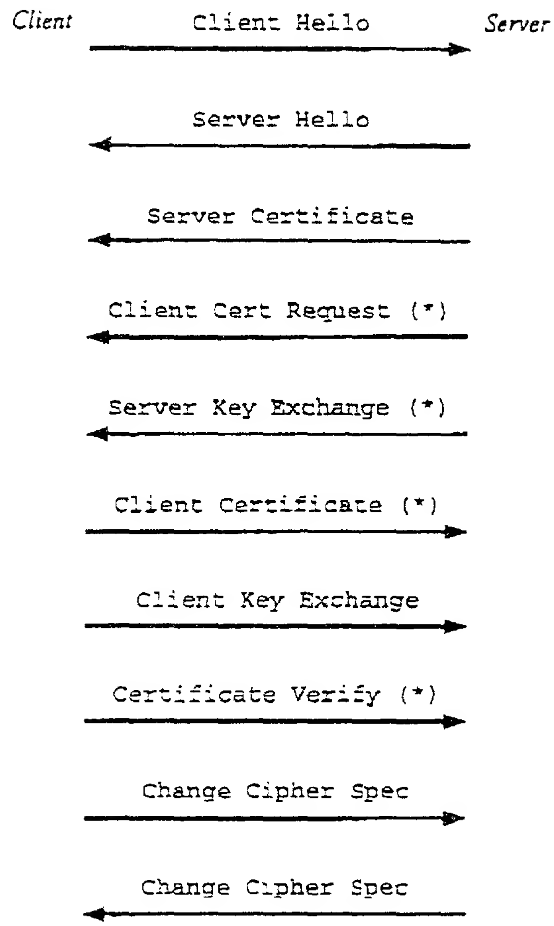

Figure 4 This ladder diagram shows the SSL negotiation/key exchange protocol. The steps marked with an asterisk are optional in the definition, and are practically never carried out in practice.

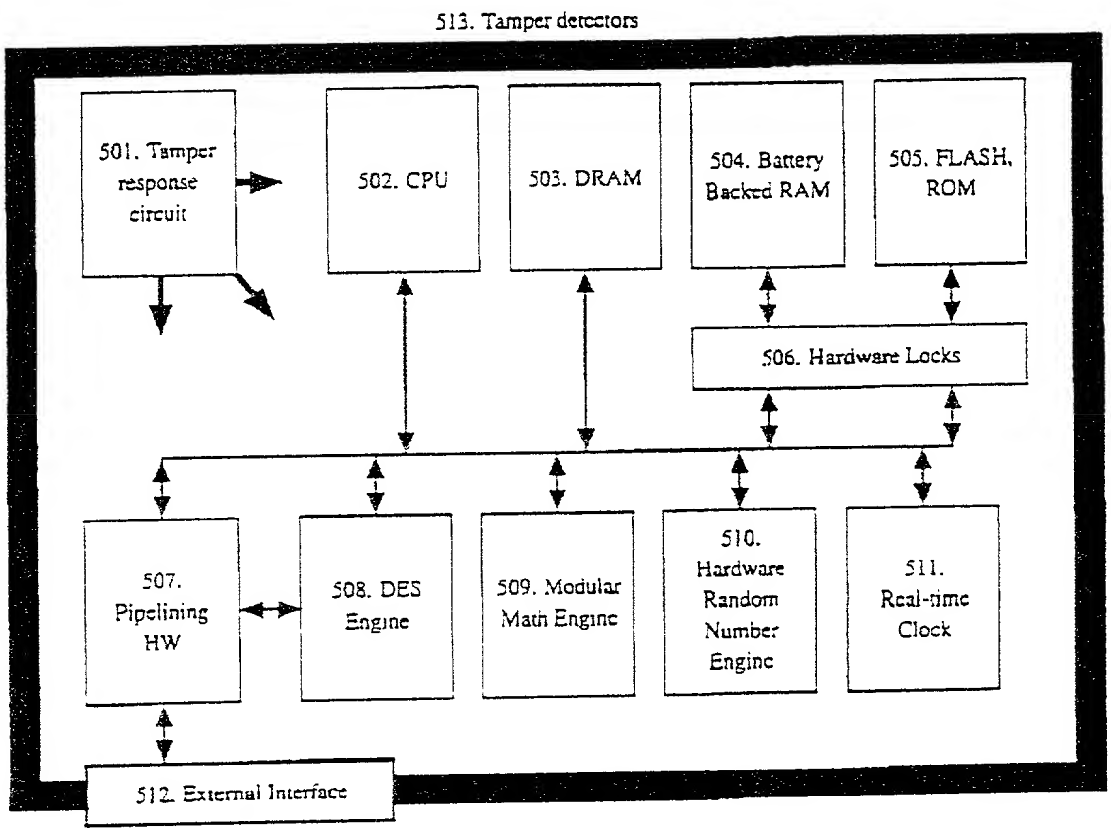

Figure 5 Exemplary Secure Coprocessor Hardware.

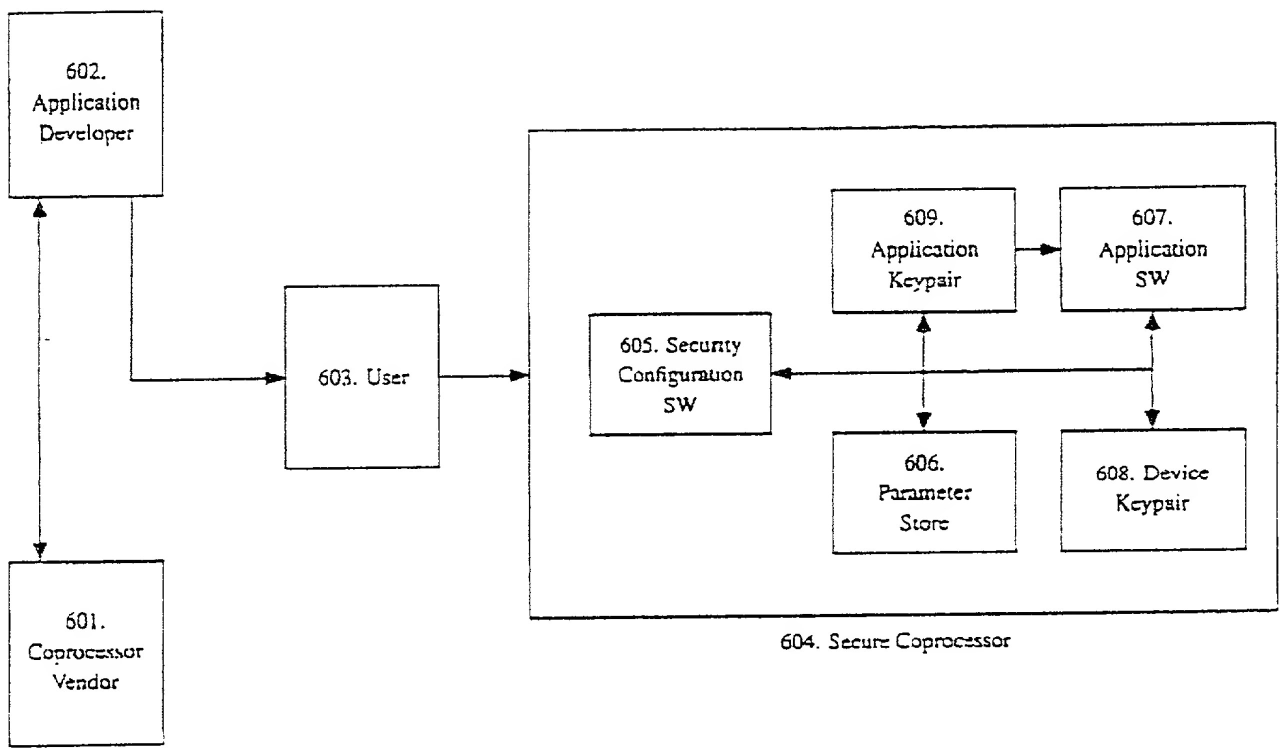

Figure 6 Exemplary Software Configuration Architecture

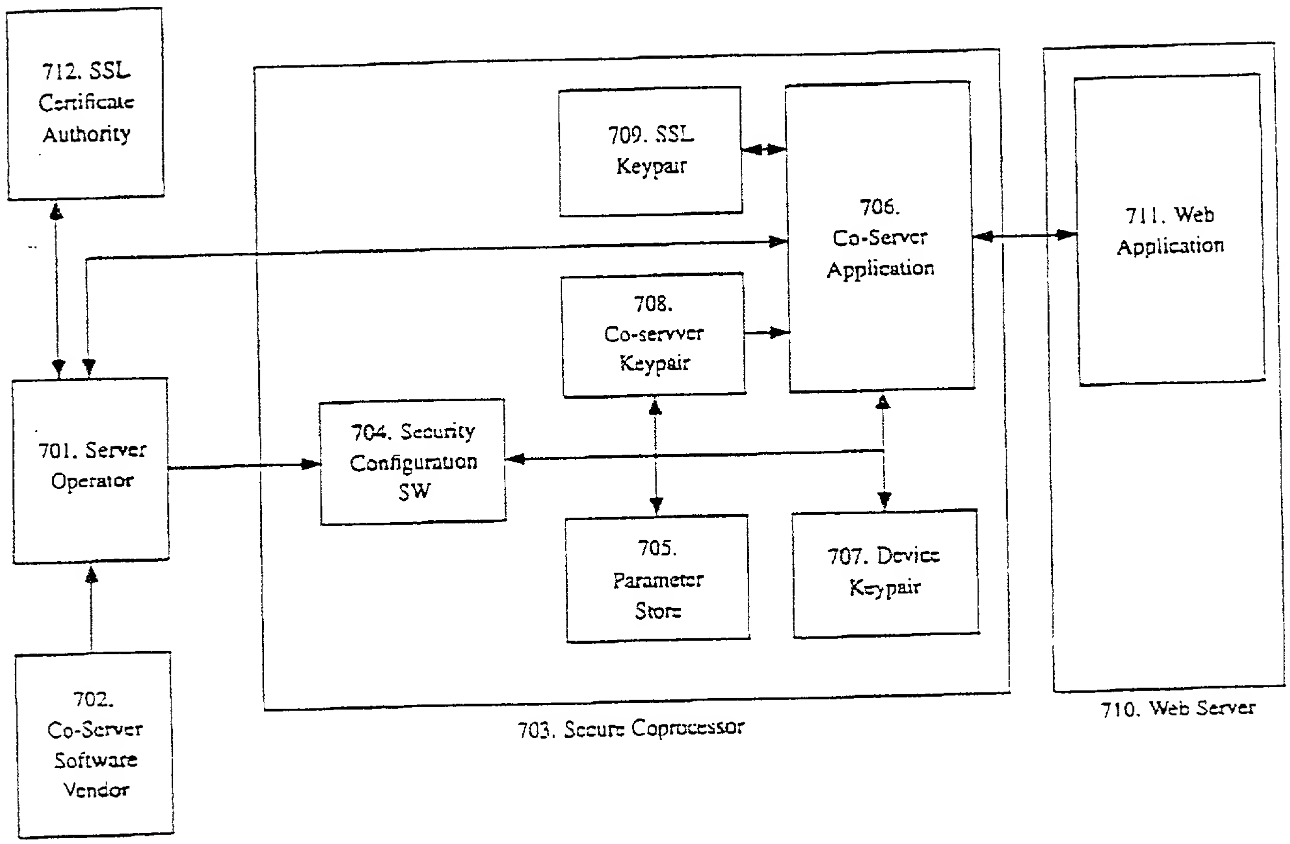

Figure 7 Exemplary Installation and Certification of Trusted Co-Server

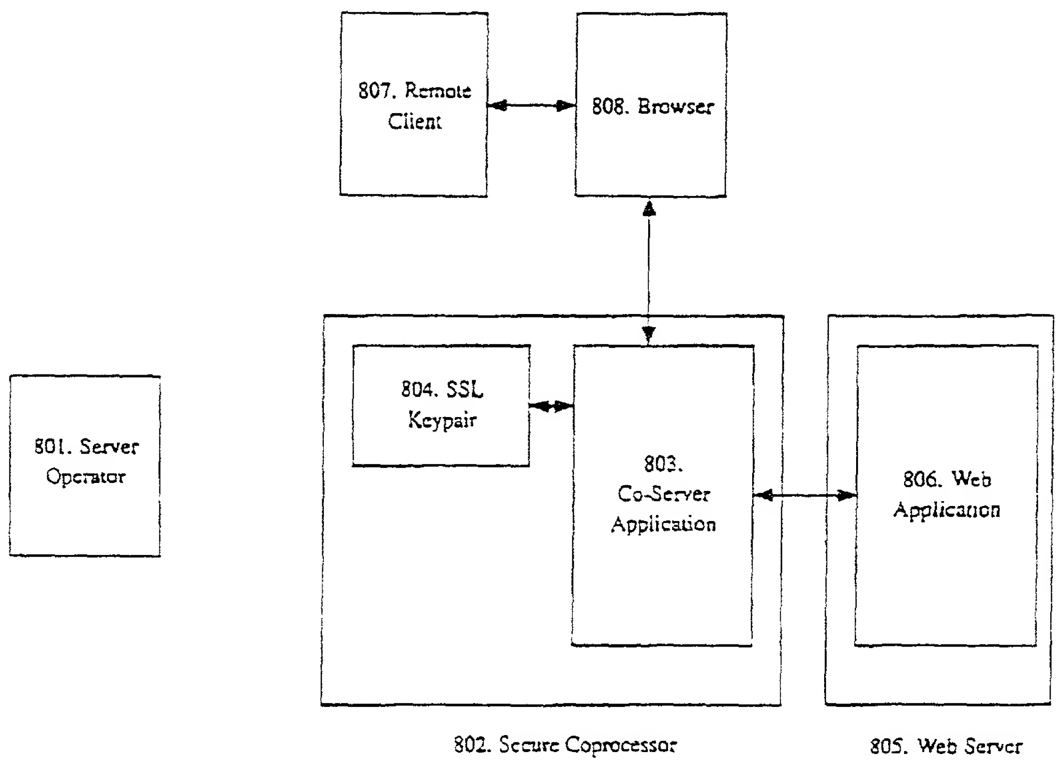

Figure 8 Exemplary establishment of secure SSL session between client and trusted co-server.

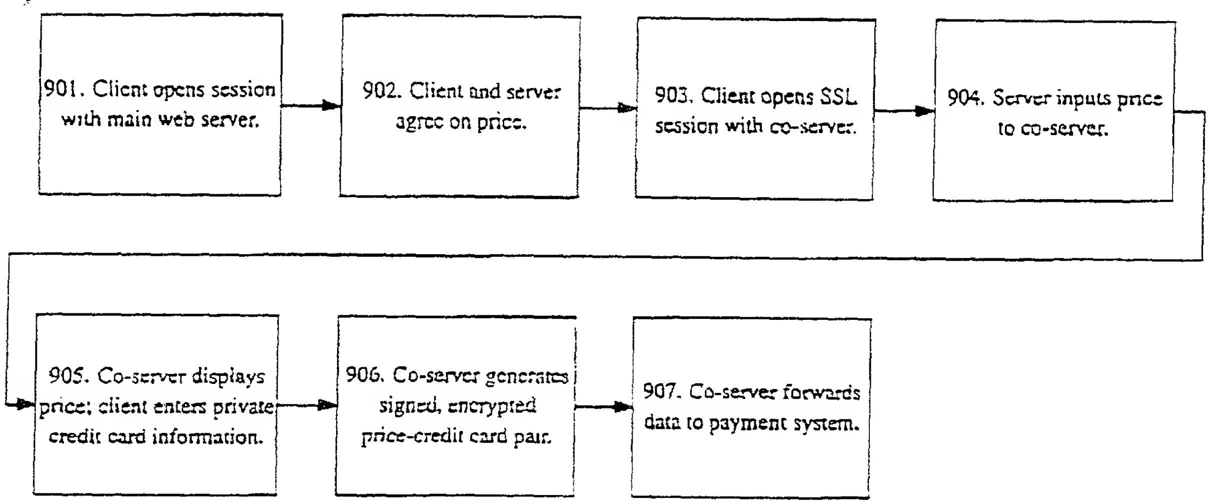

Figure 9 Example use scenario for a trusted co-server configured with a payment application.

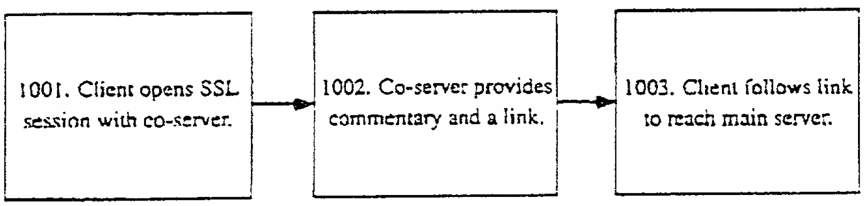

Figure 10 Example use scenario for a trusted co-server configured with a server status application.

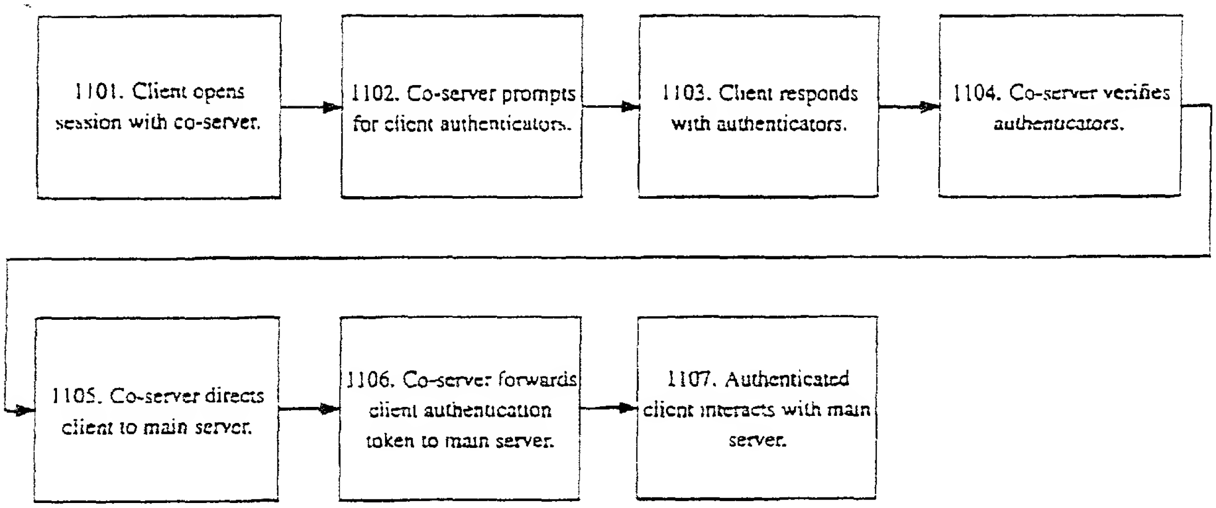

Figure 11 Example use scenario for a trusted co-server configured with an authentication application.

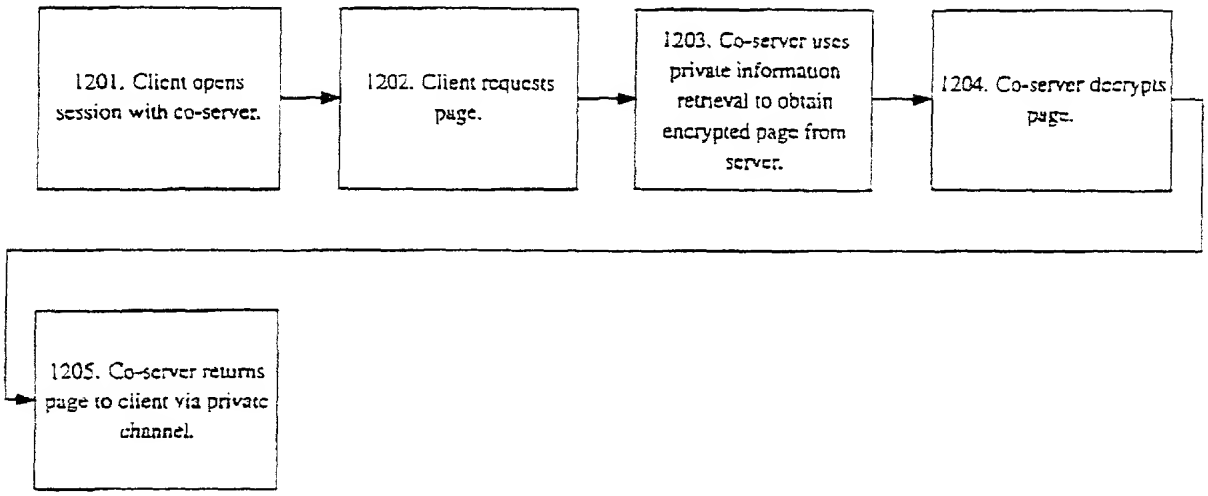

Figure 12 Example use scenario for a trusted co-server configured with a private information retrieval application.

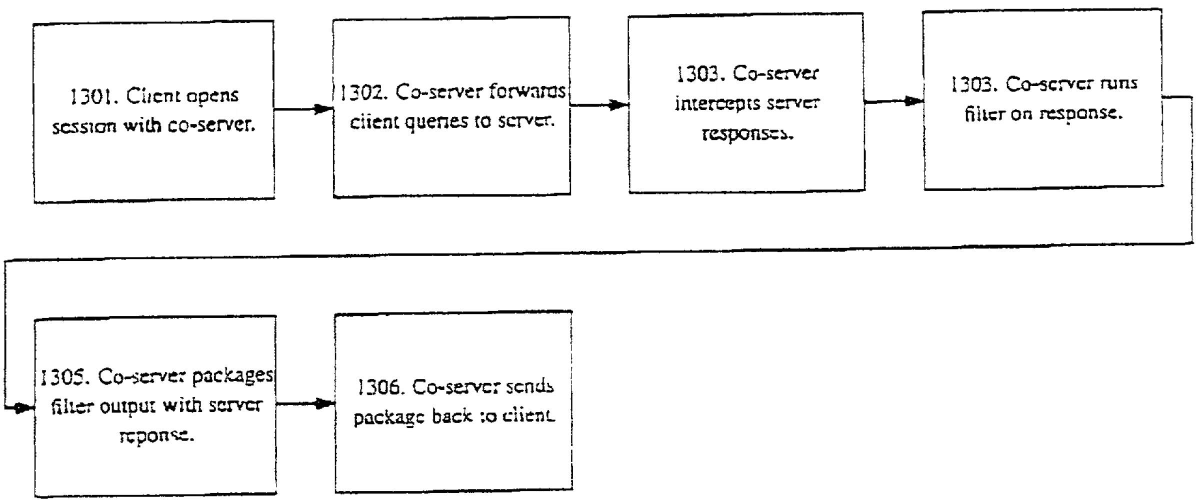

Figure 13 Example use scenario for a trusted co-server configured with a filter application.

DECLARATION AND POWER OF ATTORNEY FOR PATENT APPLICATION

As a below named inventor, I hereby declare that:

My residence, post office address and citizenship are as stated below next to my name;

I believe I am the original, first and sole inventor (if only one name is listed below) or an original, first and joint inventor (if plural names are listed below) of the subject matter which is claimed and for which a patent is sought on the invention entitled: USING TRUSTED CO-SERVERS TO ENHANCE SECURITY OF WEB INTERACTION

the specification of which (check one)

_x_ is attached hereto.

___ was filed on __________________ as United States Application Number __________________

or PCT International Application Number __________________

and was amended on __________________ (if applicable)

I hereby state that I have reviewed and understand the contents of the above identified specification, including the claims, as amended by any amendment referred to above.

I acknowledge the duty to disclose information which is material to the patentability of this application in accordance with Title 37, Code of Federal Regulations, Section 1.56.

I hereby claim foreign priority benefits under Title 35, United States Code, §119(a)-(d) or §365(b) of any foreign application(s) for patent or inventor's certificate, or §365(a) of any PCT International application which designated at least one country other than the United States, listed below and have also identified below, by checking the box, any foreign application for patent or inventor's certificate, or PCT International application, having a filing date before that of the application on which priority is claimed:

Prior Foreign Application(s)

			Priority Claimed
(Number)	(Country)	(Day/Month/Year Filed)	___ Yes ___ No
(Number)	(Country)	(Day/Month/Year Filed)	___ Yes ___ No
(Number)	(Country)	(Day/Month/Year Filed)	___ Yes ___ No

I hereby claim the benefit under 35 U.S.C. §119(e) of any United States provisional application(s) listed below.

(Application Number)	(Filing Date)
(Application Number)	(Filing Date)

I hereby claim the benefit under 35 U.S.C. §120 of any United States Application(s), or §365(c) of any PCT International application designating the United States, listed below and, insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States, or PCT International application in the manner provided by the first paragraph of 35 U.S.C. §112, I acknowledge the duty to disclose information material to the patentability of this application as defined in 37 CFR §1.56 which occurred between the filing date of the prior application and the national or PCT international filing date of this application:

(Application Serial No.)	(Filing Date)	(Status) (patented, pending, abandoned)
(Application Serial No.)	(Filing Date)	(Status) (patented, pending, abandoned)

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that willful false statements may jeopardize the validity of the application or any patent issued thereon.

POWER OF ATTORNEY: As a named inventor I hereby appoint the following attorney(s) and/or agent(s) to prosecute this application and transact all business in the Patent and Trademark Office connected therewith (list name and registration number).

Manny W. Schecter (Reg. 31,722), Lauren C. Bruzzone (Reg. No. 35,802), Christopher A. Hughes (Reg. 26,914), Edward A. Pennington (Reg. 32,588), John E. Hoel (Reg. 26,279), Joseph C. Redmond, Jr. (Reg. 18,753), Douglas W. Cameron (Reg. No. 31,596), Wayne L. Ellenbogen (Reg. No. 43,602), Stephen C. Kaufman (Reg. No. 29,551), Daniel P. Morris (Reg. No. 32,053), Louis J. Percello (Reg. No. 33,206), David M. Shofi (Reg. No. 39,835), Robert M. Trepp (Reg. No. 25,933), Paul J. Otterstedt (Reg. No. 37,411) and Louis P. Herzberg (Reg. No. 41,500) and Marian Underweiser (Reg. No. 46,134).

YOR920000457US1

SSM&P Docket No.:13807

IBM Docket No.: YOR920000457US1

DECLARATION AND POWER OF ATTORNEY FOR PATENT APPLICATION

Send Correspondence to: Richard L. Catania, Scully, Scott, Murphy & Presser

400 Garden City Plaza, Garden City, New York 11530

Direct Telephone Calls to: (name and telephone number) Richard L. Catania, (516) 742-4343

David M. Chess
Full name of sole or first inventor

Inventor's Signature Date

1744 Lawrence Road, Mohegan Lake, New York 10547
Residence

United States of America
Citizenship

Same as residence
Post Office Address

Joan Dyer
Full name of second joint inventor, if any

Inventor's signature Date

186 Riverside Drive, #5F, New York, New York 10024
Residence

United States of America
Citizenship

Same as residence
Post Office Address

Naomaru Itoi
Full name of third joint inventor, if any

Inventor's Signature Date

1647 Beal Avenue, Apt. 14, Ann Arbor, Michigan 48105-2436
Residence

Japan
Citizenship

Same as residence
Post Office Address

Jeff Kravitz
Full name of fourth joint inventor, if any

Inventor's signature Date

7 Tudor Circle, Yorktown Heights, New York 10598
Residence

United States of America
Citizenship

Same as residence
Post Office Address

Elaine Rivette Palmer
Full name of fifth joint inventor, if any

Inventor's Signature Date

293 Waccabuc Road, Goldens Bridge, New York 10526
Residence

United States of America
Citizenship

Same as residence
Post Office Address

Ronald Perez
Full name of sixth joint inventor, if any

Inventor's signature Date

65 Laurelton Road, Mount Kisco, New York 10532
Residence

United States of America
Citizenship

Same as residence
Post Office Address

Sean William Smith
Full name of seventh joint inventor, if any

Inventor's signature Date

7 Bridgeman Road, Hanover, New Hampshire 03755
Residence

United States of America
Citizenship

SSM&P Docket No.:13807

IBM Docket No.: YOR920000457US1

Same as residence

Post Office Address